Solving Linear NoGo with Combinatorial Game Theory

Haoyu Du & Martin Müller

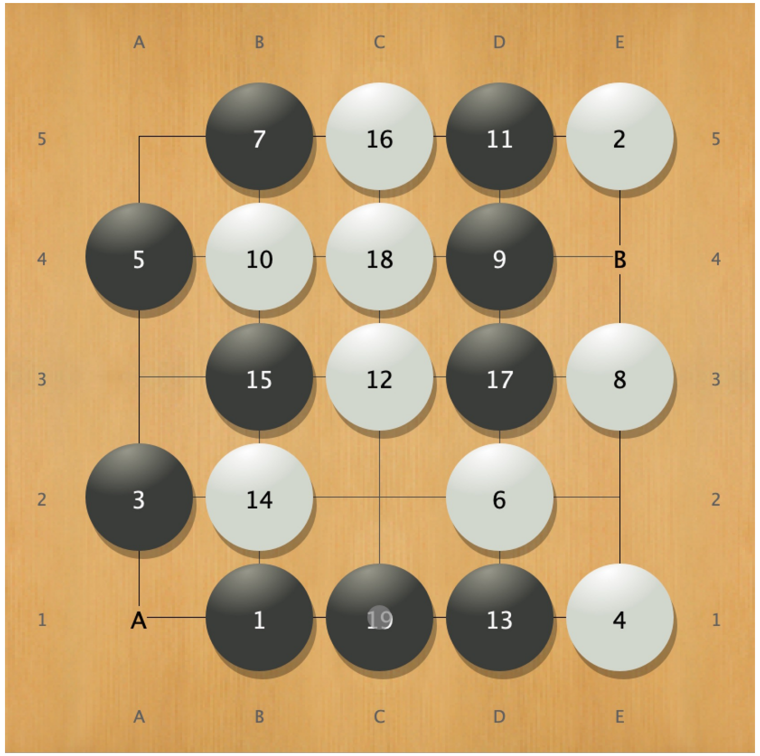
University of Alberta

NoGo

- Blocks
- Liberties

- All blocks must have at least one liberty
- Capturing and suicide are not allowed

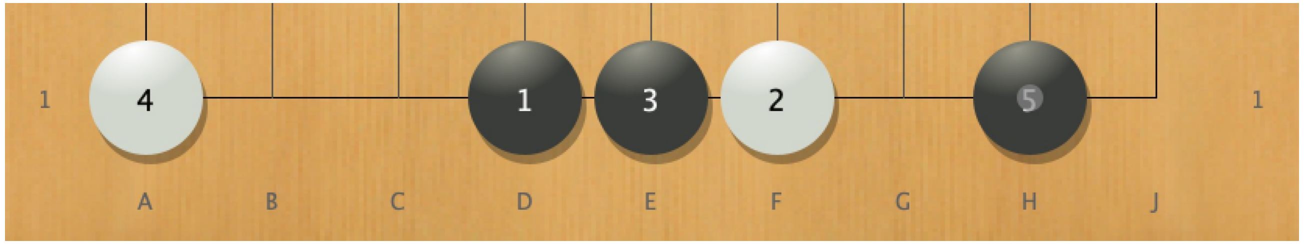
- Game ends when a player has no move to make



Linear NoGo

- NoGo played on $1 \times n$ boards
- `x` – black
- `o` – white
- `.` – empty

`o..xxo.x.`



Previous Work – CGT* Solvers

- Müller, M.: Decomposition search: A combinatorial games approach to game tree search, with applications to solving **Go endgames**. In: IJCAI. (1999)
- Song, J., Müller, M.: An enhanced solver for the game of **Amazons**. IEEE Transactions on Computational Intelligence and AI in Games 7(1), 16–27 (2015).
- Uiterwijk, J., Griebel, J.: Combining combinatorial game theory with an α-β solver for **Clobber**: theory and experiments. In: BNAIC 2016. (2017)

* CGT is short for Combinatorial Game Theory

Previous Work – NoGo

- She, P.: The design and study of NoGo program. Master’s thesis, National Chiao Tung University (2013)
- Cazenave, T.: Monte carlo game solver. In: Monte Carlo Search, MCS 2020. Communications in Computer and Information Science, vol. 1379, pp. 56–70 (2021)
- Shan, Y.C.: Solving games and improving search performance with embedded combinatorial game knowledge. Ph.D. thesis, National Chiao Tung University (2013)

Previous Work – SBHSolver

- Du, H., Wei, T.H., Müller, M.: Solving NoGo on small rectangular boards. In: Advances in Computer Games. (2023)

- SBHSolver:
- Boolean Negamax
- Sorted Bucket Hash transposition table
- Enhanced transposition cutoff
- History heuristic
- No CGT enhancements!

A New Solver for Linear NoGo

- CGTSolver: Boolean Negamax + CGT

- Block Simplification & xo-Split

- Static evaluation
- Pre-computed database
- Play-In-The-Middle (PITM) heuristic
- Transposition table

Block Simplification

Theorem 1. A block of stones of the same color can be replaced by a single stone of that color.

```
o..xxxo.xx. = o..xo.x.
```

Proof sketch. The set of legal moves does not change.

xo-Split

Theorem 2. A Linear NoGo game can be split into two independent subgames at the boundary between two blocks of opposite colors.

```
o..xxxo.xx. = o..xxx + o.xx.
```

Proof sketch. A move played on the left does not affect the liberties, and the set of legal moves, on the right.

Cancellation of Subgames

`..x.oox..xo...xoo..oxxx.`

block simplification `= ..x.ox..xo...xo..ox.`

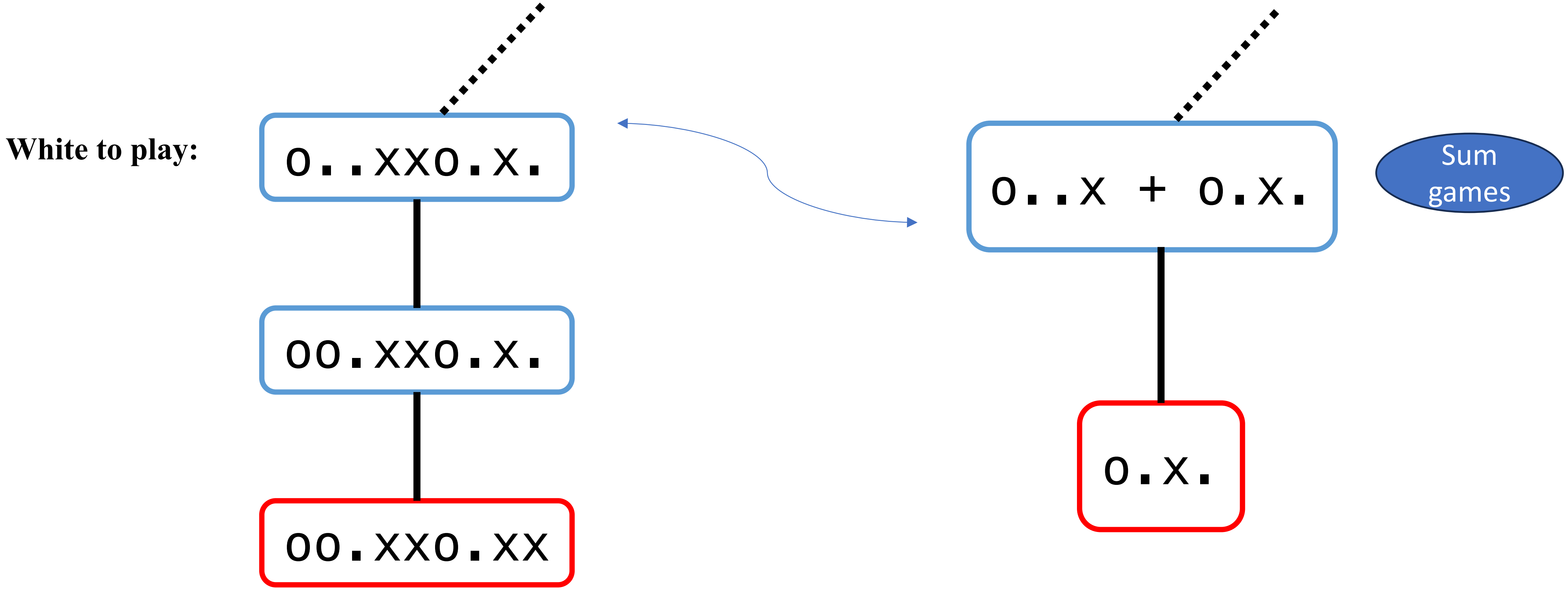
xo-split `= ..x.o + x..x + o...x + o..o + x.`

cancellation `= ..x.o + o...x + x.`

Reduced Position

Definition 1. A Linear NoGo position is called *reduced* if neither block simplification nor xo-split can be applied.

Search Trees – Classical vs CGTSolver's



Static Evaluation

$$\underbrace{\texttt{x..o.xo..ox.}}_{G} = \underbrace{\texttt{x..o.x}}_{H_1} + \underbrace{\texttt{o..o}}_{H_2} + \underbrace{\texttt{x.}}_{H_3}$$

`OutcomeClass(`H_1`) = N`	Next player wins
`OutcomeClass(`H_2`) = R`	White wins
`OutcomeClass(`H_3`) = P`	Previous player wins

`N + R + P = N + R`

If White goes next, White wins.

Pre-computed Database

Theorem 3. For all $n > 0$, there are 3^{n+1} distinct *reduced* positions with n empty points.

Proof sketch. A reduced position does not contain adjacent stones of the same color (block simplification) or opposite color (xo-split).

Reduced positions with 3 empty points: `_._._._`

```
...     .x..     o.x..     x.o.x.    o.o.o.x
```

Pre-computed Database

- Organized by layers from $n = 1$ to 15
- All reduced positions with up to 15 empty points
- $\sum_{n=1}^{15} 3^{n+1} = 64{,}570{,}077$ positions/entries
- <board, outcome class, pointer to *simplest equal game*>

Simplest Equal Games

Given a Linear NoGo position G, its simplest equal game $s(G)$ is defined to be the game that

(1) is equal to G, and

(2) appears earliest in the database.

```
..o.x.o..x = ..
```

$\underbrace{\texttt{..o.x.o..x}}_{G} = \underbrace{\texttt{..}}_{s(G)}$

- During search, replace G by $s(G)$.

Finding Simplest Equal Games

To find the simplest equal game of a position G,
for each game H_i before G in the database,
we sequentially test by search if $G - H_i = 0$.

Transposition Table

- A node is $G_1 + G_2 + \cdots$
- Encode the board of each subgame G_i as a string
- Sort board strings
- Concatenate sorted strings with separator symbols in between

s =	.	.	$\vert$	x	.	.	.	o	$\vert$	...
Black	$R^B{}_1$	$R^B{}_2$	$R^B{}_3$	$R^B{}_4$	$R^B{}_5$	$R^B{}_6$	$R^B{}_7$	$R^B{}_8$	$R^B{}_9$	...
White	$R^W{}_1$	$R^W{}_2$	$R^W{}_3$	$R^W{}_4$	$R^W{}_5$	$R^W{}_6$	$R^W{}_7$	$R^W{}_8$	$R^W{}_9$	...
Empty	$R^E{}_1$	$R^E{}_2$	$R^E{}_3$	$R^E{}_4$	$R^E{}_5$	$R^E{}_6$	$R^E{}_7$	$R^E{}_8$	$R^E{}_9$	...
Seperator	$R^{\vert}{}_1$	$R^{\vert}{}_2$	$R^{\vert}{}_3$	$R^{\vert}{}_4$	$R^{\vert}{}_5$	$R^{\vert}{}_6$	$R^{\vert}{}_7$	$R^{\vert}{}_8$	$R^{\vert}{}_9$	...

- Zobrist hashing
- $R^B, R^W, R^E, R^{\vert}$
- $h(s) = \bigoplus_i R_i^{s_i}$

Solving Linear NoGo – New Results

- 12 new results – 1×28 to 1×39
- Black wins by play the B1 opening move

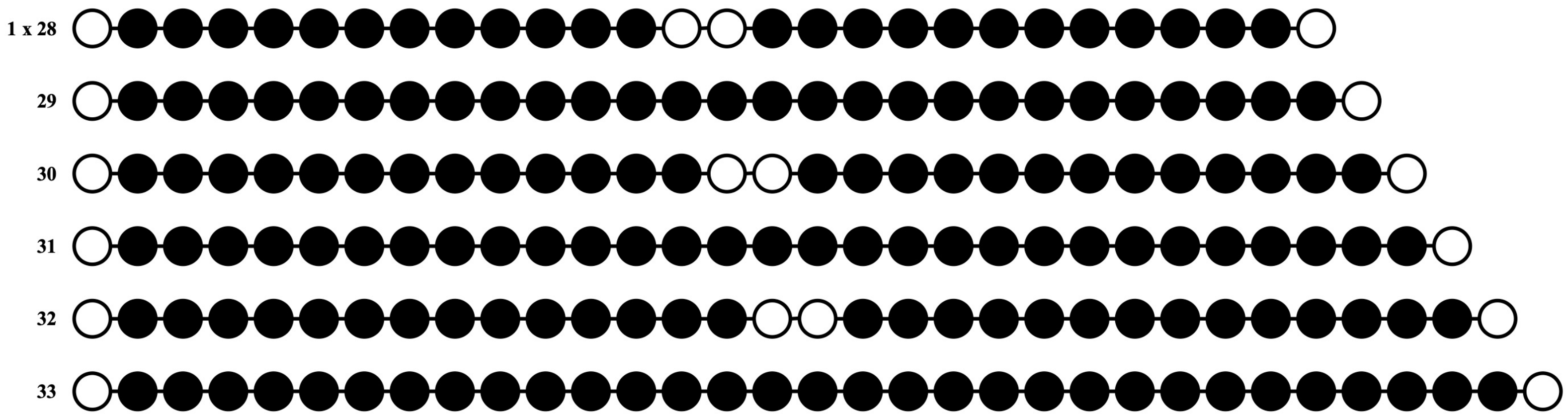
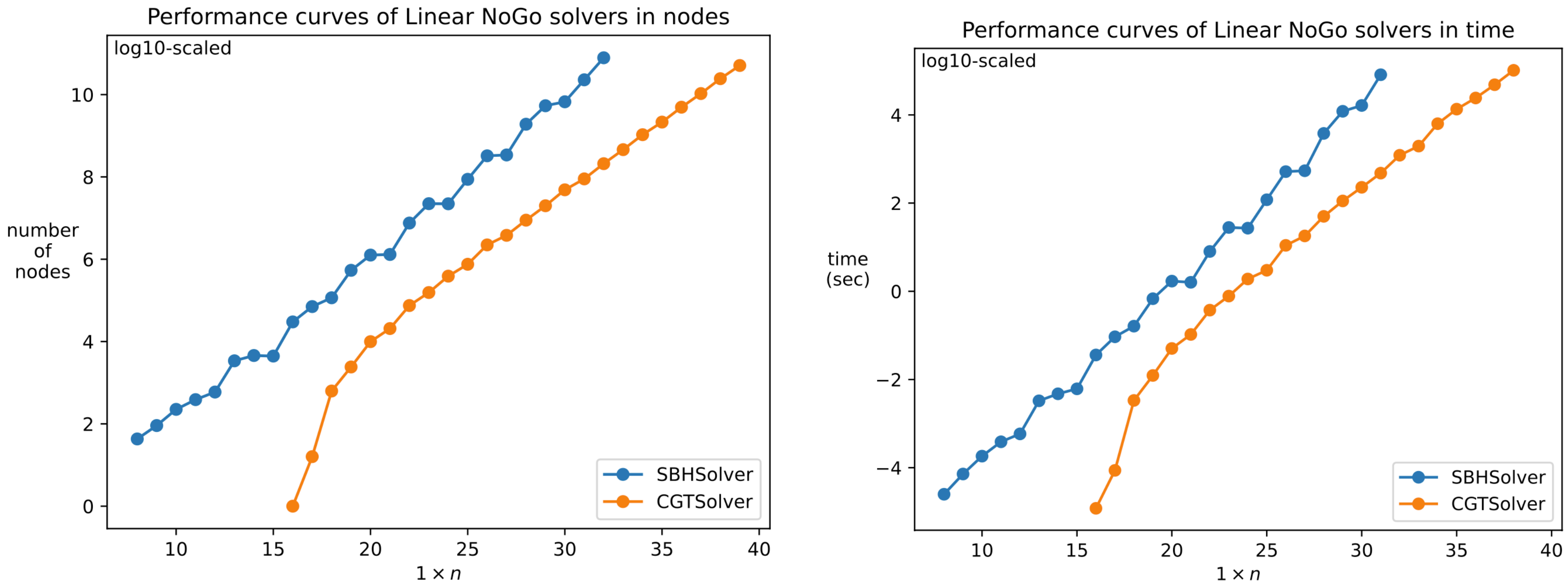


Fig. 3: The opening moves on empty $1 \times n$ NoGo with $28 \leq n \leq 33$. A black (white) stone indicates a winning (losing) opening move.

Solving Linear NoGo



(a) Node counts.   (b) Solving time.

Fig. 2: Performance comparison of SBHSolver and CGTSolver for solving $1 \times n$ NoGo with the B1 opening. The graphs are log10-scaled on the y-axis.

Ablation Study

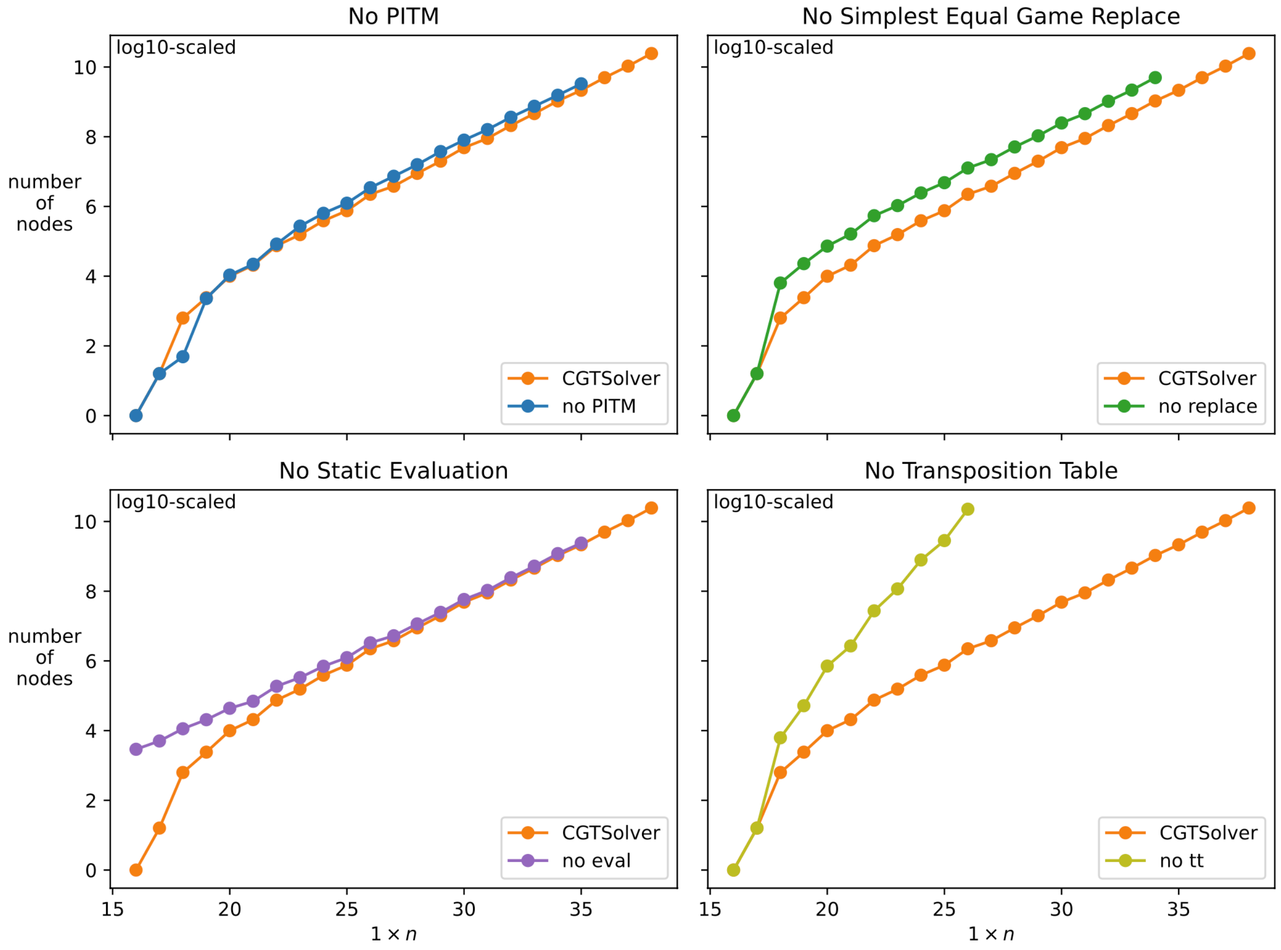


Fig. 4: Ablation on components of CGTSolver: the PITM heuristic, replacement with simplest equal games, static evaluation, and transposition table (TT).

Statistics on Search Tree and Subgames

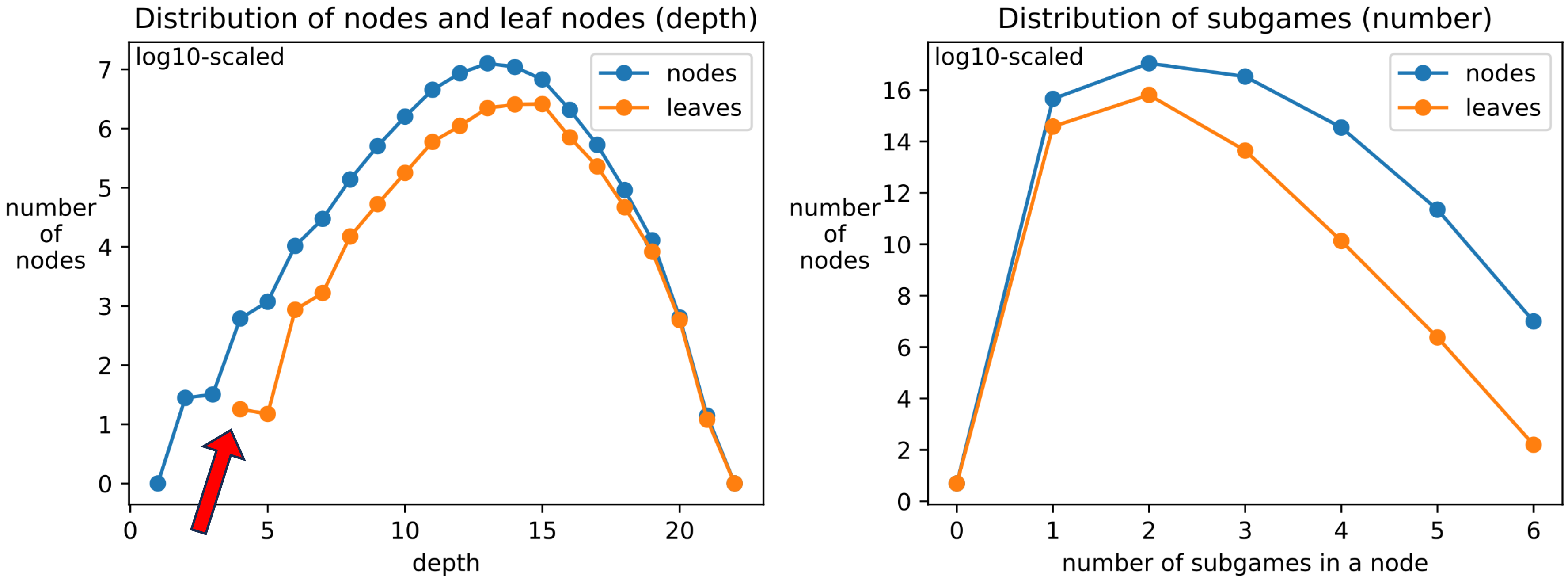


Fig. 5: Statistics in solving 1×30 NoGo with B1 opening: the number of nodes and leaf nodes across depths, the number of nodes and leaf nodes having different numbers of subgames, and the number of subgames of different sizes in all nodes

Conclusions & Future work

- 12 new results on solving Linear NoGo: board size of 28 to 39
- A CGT-enhanced solver
- More statistics and details in the paper.

- For 2D NoGo,
- Block Simplification works on any graph
- Decomposition from Shan, Y.C.: Solving Games and Improving Search Performance with Embedded Combinatorial Game Knowledge.

Occurrences of Simplest Equal Games in DB

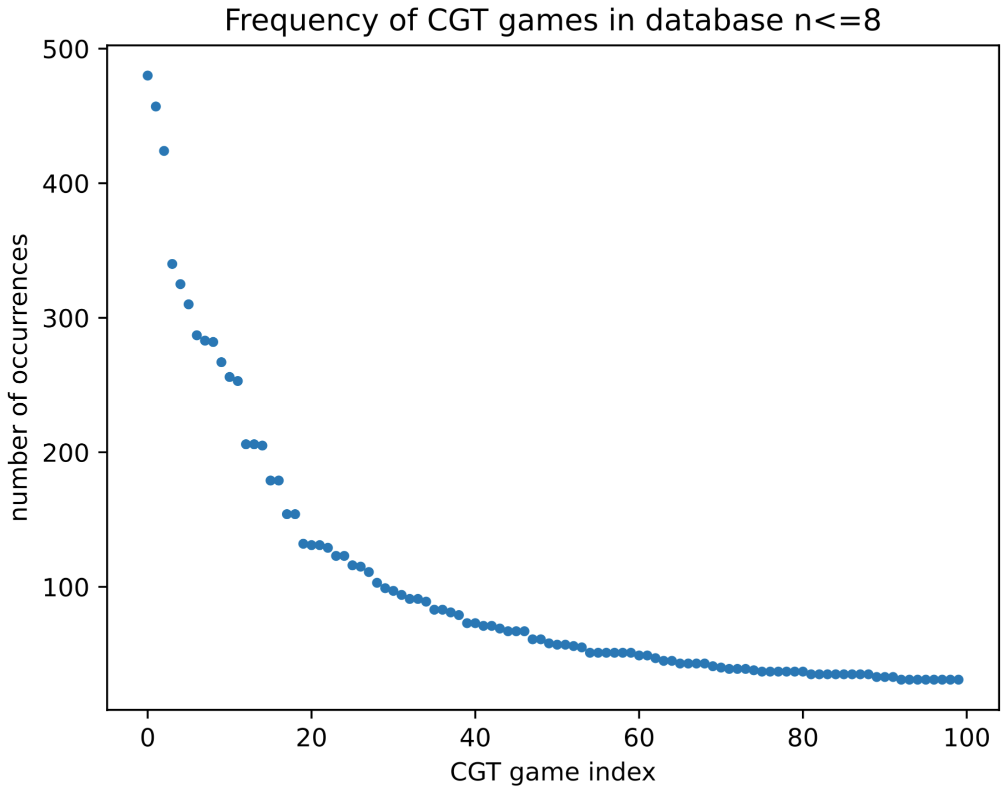


Fig. 6: The number of occurrences of the most frequent simplest equal games in the database.

idx	board	canonical form	occurrences
0	`.x.x.o.`	$1*$	481
1	`..`	$*$	457
2	`...`	± 1	424
3	`...x.o.`	$\pm(1*)$	341
4	`..x.x.`	$\{2\|1\}$	326
5	`.x.`	1	310
6	`.x.x.x.`	3	287
7	`.x.x.x.o.`	$2*$	284
8	`.`	0	282
9	`..x.`	$\{1\|0\}$	268
10	`.x.x.`	2	256

Table 1: The 11 most frequent simplest equal games in the $n \leq 8$ database.

*inverses removed

Simplest Equal Games

Due to computation constraints,

Full simplest equal games up to $n = 8$

Partial for $9 \leq n \leq 15$

idx	board	canonical form	occurrences
0	`.x.x.o.`	$1*$	481
1	`..`	$*$	457
2	`...`	± 1	424
3	`...x.o.`	$\pm(1*)$	341
4	`..x.x.`	$\{2\|1\}$	326
5	`.x.`	1	310
6	`.x.x.x.`	3	287
7	`.x.x.x.o.`	$2*$	284
8	`.`	0	282
9	`..x.`	$\{1\|0\}$	268
10	`.x.x.`	2	256

Table 1: The 11 most frequent simplest equal games in the $n \leq 8$ database.

PITM Heuristic

- Play-In-The-Middle
- Try moves closer to the middle of a subgame first.

- Move ordering:
- From largest to the smallest in length, using PITM in each subgame.

- Quickly break down a large game into smaller subgames.
- Increase the chance of database hits.

Comparison with CGSuite

- CGSuite is not efficient for solving Linear NoGo
- Needs to compute canonical forms

- 1×16 board has a massive canonical form with 1,201,194 stops!
- CGTSuite took over 8 minutes to solve.
- CGTSolver, without database, took less than 30 milliseconds.